*algorithms*

MDPI

*Article*

# An Early Investigation of the HHL Quantum Linear Solver for Scientific Applications

**Muqing Zheng *, Chenxu Liu, Samuel Stein, Xiangyu Li, Johannes Mülmenstädt, Yousu Chen and Ang Li ***

Pacific Northwest National Laboratory, Richland, WA 99354, USA; chenxu.liu@pnnl.gov (C.L.); samuel.stein@pnnl.gov (S.S.); xiangyu.li@pnnl.gov (X.L.); johannes.muelmenstaedt@pnnl.gov (J.M.); yousu.chen@pnnl.gov (Y.C.)

* Correspondence: muqing.zheng@pnnl.gov (M.Z.); ang.li@pnnl.gov (A.L.)

**Abstract**

In this paper, we explore using the Harrow–Hassidim–Lloyd (HHL) algorithm to address scientific and engineering problems through quantum computing, utilizing the NWQSim simulation package on a high-performance computing platform. Focusing on domains such as power-grid management and climate projection, we demonstrate the correlations of the accuracy of quantum phase estimation, along with various properties of coefficient matrices, on the final solution and quantum resource cost in iterative and non-iterative numerical methods such as the Newton–Raphson method and finite difference method, as well as their impacts on quantum error correction costs using the Microsoft Azure Quantum resource estimator. We summarize the exponential resource cost from quantum phase estimation before and after quantum error correction and illustrate a potential way to reduce the demands on physical qubits. This work lays down a preliminary step for future investigations, urging a closer examination of quantum algorithms' scalability and efficiency in domain applications.

**Keywords:** quantum computing; quantum simulation; hybrid software for QC-HPC



Academic Editors: Esam El-Araby and Naveed Mahmud

Received: 8 July 2025
Revised: 3 August 2025
Accepted: 5 August 2025
Published: 6 August 2025

**Citation:** Zheng, M.; Liu, C.; Stein, S.; Li, X.; Mülmenstädt, J.; Chen, Y.; Li, A. An Early Investigation of the HHL Quantum Linear Solver for Scientific Applications. *Algorithms* **2025**, *18*, 491. https://doi.org/10.3390/a18080491



## 1. Introduction

Starting with the Deutsch–Jozsa algorithm and Shor's discrete logarithm algorithm [1,2], the potential of quantum computing algorithms has extended beyond merely simulating quantum systems. The potential speedup of quantum algorithms over their classical counterparts has gathered tremendous attention, including a fundamental demand in science and engineering: solving linear systems. Harrow, Hassidim, and Lloyd (HHL) first developed a quantum linear solver with an exponential speedup in problem dimensions in [3]. Built upon the exponential speedup of quantum linear system algorithms (QLSAs), many works have explored theoretical quantum advantages in various applications. These fields include portfolio optimization [4], machine learning [5,6], differential equation solving [7], linear optimization [8–11], and semi-definite optimization [12,13].

However, the HHL algorithm proposed in [3] has a quadratic dependency on matrix condition number and matrix sparsity, worse than classical linear solvers such as factorization methods and conjugate gradient, where condition number is the product of the norm of the coefficient matrix and the norm of the inverse matrix. Several works have been proposed to reduce the dependency on the condition number of coefficient matrices and the accuracy of the solution state [14–22]. Specifically, based on adiabatic theorems,

the state of the art has a linear or quasi-linear dependency on the condition number and a logarithmic dependency on the inverse of the solution accuracy [18–20,22].

The HHL algorithm has been demonstrated in experiments to solve linear algebra problems. The largest linear systems demonstrated on real gate-based quantum machines are up to $4 \times 4$ systems with variants of the HHL algorithm [23–25] and an $8 \times 8$ system with the linear solver based on adiabatic quantum computing [26]. However, testing QLSAs on real quantum devices to demonstrate a quantum advantage still suffers from multiple obstacles, such as the large number of required quantum gates and the high noise level of current quantum devices [27].

With the current development of quantum hardware and exploration of quantum error correction (QEC) codes, a large-scale fault-tolerant quantum computer is expected to be demonstrated in the foreseeable future [28–34]. QEC codes, such as surface codes, are expected to detect and correct Pauli errors, as well as any linear combinations of them, provided the errors occur below a certain threshold probability [35]. Although the gap between algorithm requirements and hardware specifications is shrinking, the gap still exists, which necessitates the analysis of the resource costs involved [36]. Resource estimations have been performed for chemistry [37], Grover's algorithm on the Advanced Encryption Standard [38], Shor's discrete logarithm algorithm for the RSA cryptosystem [39], and the computation of elliptic curve discrete logarithms [40]. However, despite this being essential for understanding the disparity between hardware capabilities and practical applications, there is limited work on non-asymptotic resource estimation for QLSAs [41].

In this paper, we focus on resource estimation and experiment with the HHL algorithm on several applications selected from domain science, such as power grid and climate projection. Different from the previous works about asymptotic and non-asymptotic resource analysis [3,14–22,41], we investigate the factors affecting the final accuracy, resource cost, and fault-tolerant hardware requirements. Our experiments show the effectiveness of the HHL algorithm in scientific applications with a low accuracy in quantum phase estimation. Working with the Microsoft Azure Quantum resource estimator [42,43], we summarize the exponential dependency of quantum resources on the number of clock qubits in HHL circuits and demonstrate a possible method to reduce the demands on physical qubits in fault-tolerant quantum computing.

This paper is organized as follows: Section 2 introduces the idea of quantum linear system solvers, with implementation-related details. Section 3 presents the simulator, NWQSim [44], and the resource estimation tool. Next, we explore the factors of interest in evaluating numerical experiments in Section 4 and perform those experiments in Section 5. Finally, we discuss the limitations in Section 6 and conclude the implications of our work on domain science applications in Section 7.

## 2. Quantum Linear Systems and the Implementation of the Solver

### *2.1. Overview of the Harrow–Hassidim–Lloyd (HHL) Algorithm*

Quantum information is encoded into the state of quantum systems. Here, we assume all relevant quantum states can be represented as statevectors. An $n_d$-qubit statevector $|x\rangle = \sum_{j=0}^{2^{n_d}-1} \alpha_j |\vec{j}\rangle$ is a normalized complex vector, i.e., $\alpha_j \in \mathbb{C}$ for all $j$ and $\sum_{j=0}^{2^{n_d}-1} |\alpha_j|^2 = 1$, while $\vec{j} \in \{0,1\}^{n_d}$ is the number $j$ as a binary string. The set $\{|\vec{j}\rangle\}$ forms the basis set of $\mathbb{C}^{2^{n_d}}$, referred to as the computational basis. Specifically, $|\vec{j}\rangle$ is the unit vector whose $(j+1)^{\text{th}}$ entry is 1 and other entries are 0. The notation $\langle\vec{j}|$ is the conjugate transpose of $|\vec{j}\rangle$.

**Definition 1** (A quantum linear-system problem)**.** *A quantum linear-system problem is to solve a system of linear equations with a normalized solution vector* $|x\rangle = A^{-1}|b\rangle / \|A^{-1}|b\rangle\|_2$ *where coefficient matrix* $A \in \mathbb{C}^{N\times N}$ *is Hermitian and* $|x\rangle$ *and* $|b\rangle$ *are both normalized vectors.*

Start with a classical complex linear system $A\vec{x} = \vec{b}$, $A \in \mathbb{C}^{N\times N}$, where the right-hand-side (RHS) vector $\vec{b}$ is normalized to obtain $|b\rangle := \vec{b}/\|\vec{b}\|_2$; then, a quantum linear-system problem can be formed with $A$ and $|b\rangle$ if $A$ is Hermitian. Otherwise, a larger linear system can be constructed as follows [3]:

$$\begin{bmatrix} \mathbf{0} & A \\ A^\dagger & \mathbf{0} \end{bmatrix} \begin{bmatrix} \vec{0} \\ |x\rangle \end{bmatrix} = \begin{bmatrix} |b\rangle \\ \vec{0} \end{bmatrix}, \tag{1}$$

where $\cdot^\dagger$ is the conjugate transpose. Therefore, we assume coefficient matrices are Hermitian in the rest of this paper. Since the data is encoded into qubits, if the dimensions of $A$ and $\vec{b}$ are not in the power of 2, $A$ and $b$ must be expanded. Suppose there exists a quantum linear system solver that obtains $|x\rangle = A^{-1}|b\rangle / \|A^{-1}|b\rangle\|_2$ from the circuit; then the original solution of the system can be recovered by $\vec{x} = \|b\|_2 \|A^{-1}|b\rangle\|_2 |x\rangle$. While $\|b\|_2$ is known from the previous computation, the solver needs to provide the value of $\|A^{-1}|b\rangle\|_2$.

#### 2.1.1. Mathematical Foundation of HHL

Harrow, Hassidim, and Lloyd first developed the HHL algorithm to solve the quantum linear-system problem [3]. The fundamental idea behind the HHL algorithm is that the eigenstates of the Hermitian matrix $A$ (noted as $\{|v_j\rangle\}$) form a complete orthonormal basis of $\mathbb{C}^N$ (i.e., $\langle v_j|v_k\rangle = \delta_{jk}$), and hence the state $|b\rangle$ can always be decomposed by this basis as $|b\rangle = \sum_{j=0}^{N-1} b_j |v_j\rangle$. Similarly,

$$\begin{aligned} |x\rangle &= \frac{A^{-1}|b\rangle}{\|A^{-1}|b\rangle\|_2} \\ &= \frac{1}{\|A^{-1}|b\rangle\|_2} \sum_{j=0}^{N-1} \frac{1}{\lambda_j} |v_j\rangle \langle v_j| \sum_{j=0}^{N-1} b_j |v_j\rangle \\ &= \frac{1}{\sqrt{\sum_{j=0}^{N-1} \frac{|b_j|^2}{\lambda_j^2}}} \sum_{j=0}^{N-1} \frac{b_j}{\lambda_j} |v_j\rangle . \end{aligned} \tag{2}$$

In other words, the HHL algorithm needs a quantum computer to perform eigen-decomposition of $A$ and eigenvalue inversion. Figure 1 shows a general description of the circuit that exactly serves the purpose, with an additional $n_d$-qubit data-loading block to load $|b\rangle$ into the quantum computer and $n_d = \lceil \log(N) \rceil$.

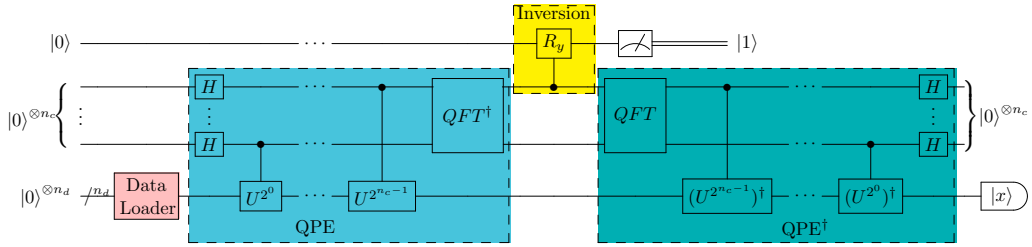


**Figure 1.** HHL circuit. The unitary gates in quantum phase estimation (QPE) are $U = e^{itA}$ and $U^{2^j} = e^{i2^j tA}$ where $i^2 = -1$ and $t$ is a scaling factor. The top qubit is referred to as the ancillary qubit, and it is the most significant qubit.

#### 2.1.2. Quantum Phase Estimation

The eigen-decomposition requires a subroutine called quantum phase estimation (QPE), as illustrated in the sky blue part of Figure 1. Given a unitary matrix $U$ has an eigenstate $|v_j\rangle$ with eigenvalue $e^{2\pi i\theta_j}$, QPE is a quantum algorithm to solve the phase of the eigenvalue ($\theta_j$) [35]. After executing the QPE algorithm, the binary representation of

the phase angle $\theta$ is stored in a qubit state. The qubits carrying the phase information are named "clock qubits". In the HHL algorithm, if $|v_j\rangle$ is an eigenstate of a Hermitian matrix $A$ with eigenvalue $\lambda_j$, by constructing a unitary matrix $U = e^{itA}$ with a scale factor $t$, the state $|v_j\rangle$ becomes an eigenstate of $U$ with eigenvalue $e^{it\lambda_j}$. Therefore, the eigenvalue $\lambda_j$ can be estimated using the QPE algorithm.

Suppose we have access to the gate $U$; then it is clear that $U^l |v_j\rangle = e^{2\pi i\theta_j l} |v_j\rangle$ for some positive integer $l$. QPE requires a submodule called quantum Fourier transform (QFT). QFT maps

$$\begin{aligned} QFT\,|\vec{j}\rangle &= \frac{1}{\sqrt{2^{n_c}}} \sum_{k=0}^{2^{n_c}-1} \omega^{jk} |\vec{k}\rangle \\ &= \frac{1}{\sqrt{2^{n_c}}} \left(|0\rangle + e^{2\pi i 0.j_{n_c}} |1\rangle\right) \left(|0\rangle + e^{2\pi i 0.j_{n_c-1}j_{n_c}} |1\rangle\right) \cdots \left(|0\rangle + e^{2\pi i 0.j_1 j_2 \ldots j_{n_c}} |1\rangle\right) \end{aligned}$$

where $\vec{j}$ has $n_c$ bits and $\omega = e^{2\pi i/(2^{n_c})}$. Intuitively, $\vec{j}$ can be considered as binary number $\vec{j} = j_1 j_2 \ldots j_{n_c}$ such that $j_l \in \{0, 1\}$ and QFT transforms this binary number from a state to the phases of bases in different accuracy. So, on the contrary, if we apply the inverse of the QFT operator, denoted by $QFT^\dagger$, the phase value becomes a state, and we can measure the state to obtain the phase value in the binary representation.

To summarize the process of a standalone QPE routine, we have

$$\begin{aligned} |0\rangle^{\otimes n_c} |v_j\rangle &\xrightarrow{H^{\otimes n_c}} \frac{1}{\sqrt{2^{n_c}}} \sum_{k=0}^{2^{n_c}-1} |\vec{k}\rangle |v_j\rangle \\ &\xrightarrow{CU \text{ sequence}} \frac{1}{\sqrt{2^{n_c}}} \sum_{k=0}^{2^{n_c}-1} e^{2\pi i\theta_j k} |\vec{k}\rangle |v_j\rangle \\ &\xrightarrow{QFT^\dagger} |\tilde{\theta}_j\rangle |v_j\rangle \end{aligned}$$

where the $CU$ sequence is the controlled-$U$ sequence in the sky blue part of Figure 1 and $\tilde{\theta}_j = \theta_j$ if $\theta_j$ can be perfectly represented in $n_c$ bits; otherwise, $\tilde{\theta}_j$ is an estimation of $\theta_j$ in a finite accuracy. In other words, the number of clock qubits, $n_c$, governs the accuracy of the estimated eigenvalue in QPE. To understand more details about QFT and QPE, we direct the interested reader to [35,45].

Note that, without circuit optimization, the increase in $n_c$ will exponentially increase the gate counts in HHL circuits. Recall the HHL circuit in Figure 1: an extra clock qubit leads to an extra controlled $U^{2^{n_c-1}}$ and an extra controlled inverse $U^{2^{n_c-1}}$ in the HHL circuit, where $U = e^{itA}$ and $A$ is the coefficient matrix in a linear system. Generally, we should not explicitly compute the matrix $U^{2^{n_c-1}}$, but apply gate $U$ for $2^{n_c-1}$ times in the circuit. Then, the QPE part of the circuit contains $\sum_{j=0}^{n_c-1} 2^j = 2^{n_c} - 1$ number of $U$, and when there are $n_c + 1$ clock qubits, an extra $2^{n_c}$ number of $U$ is added into the QPE, which almost doubles the number of gates $U$ in QPE. The same situation happens on the inverse QPE part of the HHL circuit.

#### 2.1.3. State Evolution in HHL

In general, the evolution of states in the HHL circuit is

$$
\begin{aligned}
|0\rangle |0\rangle^{\otimes n_c} |0^{n_d}\rangle &\xrightarrow{\text{Data loading}} |0\rangle |0\rangle^{\otimes n_c} |b\rangle \\
&\xrightarrow{\text{QPE}} \sum_{j=0}^{2^{n_d}-1} b_j |0\rangle |\tilde{\lambda}_j\rangle |v_j\rangle \\
&\xrightarrow[\text{Inversion}]{\text{Eigenvalue}} \sum_{j=0}^{2^{n_d}-1} b_j \left( \sqrt{1 - \frac{C^2}{\tilde{\lambda}_j^2}} |0\rangle + \frac{C}{\tilde{\lambda}_j} |1\rangle \right) |\tilde{\lambda}_j\rangle |v_j\rangle \\
&\xrightarrow[\text{only keep } |1\rangle]{\text{Measure the ancillary}} D \sum_{j=0}^{2^{n_d}-1} \frac{b_j}{\tilde{\lambda}_j} |1\rangle |\tilde{\lambda}_j\rangle |v_j\rangle \\
&\xrightarrow{\text{QPE}^\dagger} D |1\rangle |0^{n_c}\rangle \sum_{j=0}^{2^{n_d}-1} \frac{b_j}{\tilde{\lambda}_j} |v_j\rangle \approx D |1\rangle |0^{n_c}\rangle |x\rangle
\end{aligned}
$$

where $\tilde{\lambda}_j \approx \lambda_j$ is the eigenvalue of $A$ with a finite accuracy estimated in QPE, and $C$ and $D$ are both constant normalization factors

$$D = \frac{C}{\sqrt{\sum_{j=0}^{2^{n_d}-1} C^2 \frac{|b_j|^2}{\tilde{\lambda}_j^2}}} \approx \frac{1}{\| A^{-1} |b\rangle \|_2}.$$

Thus, the norm $\| A^{-1} |b\rangle \|_2$ can be estimated by the probability of measuring the ancillary qubit in state $|1\rangle$, i.e., $\| A^{-1} |b\rangle \|_2 \approx \sqrt{\Pr(\text{Measure 1 in ancillary})}$. This value can be obtained without extra cost as we need to run the circuit multiple times to get $|x\rangle$ or $\langle x| M |x\rangle$ for some observable $M$. The overall runtime complexity of HHL algorithm is $\tilde{O}(\log(N) s^2 \kappa^2 / \epsilon)$, where $s$ is the sparsity of $A$, $\kappa = \|A\| \|A^{-1}\|$ is the condition number of $A$, and $\epsilon$ is the final additive error of the solution defined by the ideal state $|x\rangle$ and the result from HHL $|x_{HHL}\rangle$ through $\| |x\rangle - |x_{HHL}\rangle \| \leq \epsilon$ [3].

#### 2.1.4. Quantum-Classical Data Exchange in HHL

There are two major input models for encoding both matrix $A$ (or $e^{itA}$) and vector $|b\rangle$ into a quantum computer. One is the sparse-access model, used in the HHL algorithm [3]. The sparse-access model is a quantum version of classical sparse matrix computation, and we assume access to unitaries that calculate the index of the $l^{\text{th}}$ non-zero element of the $k^{\text{th}}$ row of a matrix $A$ when given $(k, l)$ as input. A different input model, now known as the quantum operator input model, is from Low and Chuang [46]. This method is based on the block-encoding of $A$ to allow efficient access to entry values. Its circuit implementation can be found in [47,48]. Meanwhile, this encoding scheme can also be achieved using quantum random access memory [49–52]. It requires the complexity $O(\text{polylog}(N/\epsilon_{BE}))$ for realizing an $\epsilon_{BE}$-approximate block-encoding of $A \in \mathbb{C}^{N \times N}$ with quantum random access memory [50].

**Definition 2** (The block-encoding of a matrix)**.** *The block-encoding of a matrix $A \in \mathbb{C}^{N \times N}$ is a unitary operator $U$ such that*

$$U = \begin{bmatrix} A/a & \cdot \\ \cdot & \cdot \end{bmatrix}$$

*where $a \geq \|A\|$ is a normalizing constant. In other words, $U$ and $A$ satisfy, for some constant $a$ and $n$,*

$$a\left(\langle 0|^{\otimes n} \otimes I_N\right) U \left(I_N \otimes |0\rangle^{\otimes n}\right) = A$$

*where $I_N \in \mathbb{R}^{N\times N}$ is the identity matrix.*

To construct the matrix exponential $e^{itA}$ in QPE via block-encoding-based methods, block-encoding first embeds the coefficient matrix $A$ into a larger unitary matrix, and then uses Quantum Signal Processing to achieve the Hamiltonian simulation $e^{itA}$ [53]. However, limited by the size of tested matrices in our numerical experiments, block-encoding alone already does not show advantages over Quantum Shannon Decomposition (QSD), a method that directly decomposes $e^{itA}$ into a sequence of basis gates, in terms of gate counts and the number of encoding qubits. We illustrate this in Table 1. Note that the coefficient matrix used in Section 5.1 is non-Hermitian, so its conversion to a Hermitian matrix using Equation (1) brings a disadvantage to QSD in terms of the number of gates.

**Table 1.** Resource costs of using FABLE [47] to block-encode $A$ and QSD to synthesize $e^{2\pi iA}$.

| Linear System | Problem Dimension | Method | # of Qubits | # of $CX$ | # of $U_3$[1] |
|---|---|---|---|---|---|
| Section 5.1 Iter. 1 | $5 \times 5$ | QSD | 4 | 100 | 208 |
| | | FABLE | 7 | 73 | 70 |
| Section 5.2 Three-point | $9 \times 9$ | QSD | 4 | 100 | 208 |
| | | FABLE | 9 | 268 | 264 |
| Section 5.2 Five-point | $25 \times 25$ | QSD | 5 | 444 | 904 |
| | | FABLE | 11 | 1039 | 1034 |

[1] $U_3$ gate is the 1-qubit rotation gates with 3 Euler angles.

On the other hand, efficiency in reading $|x\rangle$ could be a potential threat to quantum speedup. The current state-of-the-art quantum state tomography algorithm is from Apeldoorn et al. [54]. For a state $|\psi\rangle = \sum_{j=0}^{N-1} \beta_j |\vec{j}\rangle \in \mathbb{C}^N$, with probability $1-\delta$, the pure-state tomography in [54] requires $O\left(\sqrt{N}/\sigma \cdot \log(N/\delta)\right)$ queries to the unitary oracle that prepares $|\psi\rangle$ from $|00\ldots0\rangle$ to output a vector $\vec{\beta}_{est} \in \mathbb{R}^N$ such that $\|\Re(\vec{\beta}) - \vec{\beta}_{est}\|_\infty \leq \sigma$. The same routine can be applied to $i|\psi\rangle$ to estimate the imaginary part.

### *2.2. Implementation of the Circuit Generation*

In all experiments in this paper, the code for the HHL circuit generation comes from a Qiskit-based open-sourced package [55], which only produces the essential parts of the HHL circuit as colored in Figure 1. We made slight modifications to accommodate the changes in Qiskit 0.46. The state preparation for $|b\rangle$ uses the algorithm in [56] that decomposes an arbitrary isometry into the optimized number of single-qubit and CNOT gates, where isometry refers to the inner-product-preserving transformation that maps between two Hilbert spaces; i.e., the state preparation is a special case of isometries. To construct the unitary operator $e^{itA}$ in the QPE stage, the code directly uses Quantum Shannon Decomposition to synthesize the matrix exponential in the circuit.

## 3. Simulator and Resource Estimation Tool

The statevector simulator carries the simulations in the experiments, SV-Sim [57], in Northwest Quantum Circuit Simulation Environment (NWQSim)(V2.5.0) [44]. As shown in [57], compared to simulators in Aer from Qiskit [58] and qsim from Cirq [59], NWQSim provides specialized computation for a wide range of supported basis gates and archi-

tectures of CPUs and GPUs, such as gate fusion. In Table 2 and later in Section 5, we demonstrate that the gate fusion strategy in NWQSim can reduce about 80% of gates in the circuits without sacrificing error rates. On the other hand, NWQSim utilizes a communication model called "PGAS-based SHMEM" that significantly reduces communication latency for intra-node CPUs/GPUs and inter-node CPU/GPU clusters. In this case, SV-Sim has an exceptional performance over other simulators in deep-circuit simulation [57]. Figure 2 shows the running time of the HHL circuit in the size of 11 qubits to 17 qubits on SV-Sim on four different GPUs.

**Table 2.** HHL circuit properties for four random examples.

| No. of Qubits | | CX Gates | Depth | Total No. of Gates | |
|---|---|---|---|---|---|
| $n_d$ | $n_c$ | | | Before Fusion | After Fusion |
| 4 | 6 | 116,535 | 248,084 | 325,189 | 70,804 |
| 5 | 7 | 1,111,178 | 2,373,842 | 3,106,244 | 665,921 |
| 6 | 8 | 9,335,345 | 19,969,964 | 26,117,061 | 5,557,777 |
| 7 | 9 | 78,420,632 | 167,816,254 | 219,386,270 | 46,631,320 |

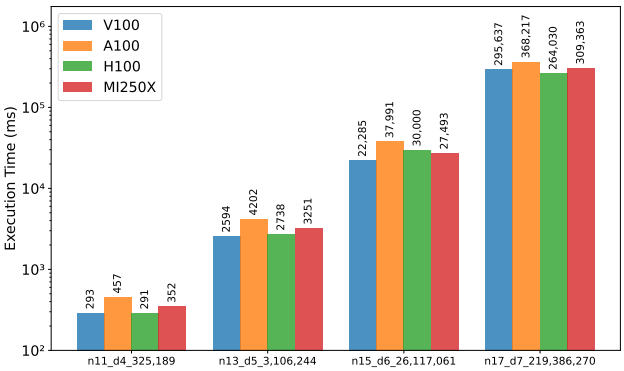


**Figure 2.** NWQSim performance on different GPUs. The testing HHL circuits use randomly generated sparse matrices and random RHS vectors. The three numbers in the name of each testing circuit are the number of qubits in the circuit, the number of qubits for data loading, and the total number of gates in the circuit, respectively.

The resource estimator in [42,43] from Microsoft Azure Quantum establishes a systematic framework to access and model the resources necessary for implementing quantum algorithms on a user-specified fault-tolerant scenario. This tool enables detailed estimation of various computational resources, such as the number of physical qubits, the runtime, and other QEC-related properties to achieve a quantum advantage for certain applications. Specifically, the tool accepts a wide range of qubit and quantum error correction code specifications and an error budget that allows different error rates to simulate a described fault-tolerant environment.

The tool is compatible with circuits generated from a high-level quantum computing language or package, including Qiskit and Q#. After a circuit is given, the input is compiled into Quantum Intermediate Representation through a unified processing program, and the estimator can examine the code and record qubit allocation, qubit release, gate operation, and measurement operation. Then, logical-level resources are estimated and used to compute the required physical-level resources further. The tool returns a thorough report on resources demanded to perform the given algorithm on fault-tolerant quantum computers, including the explanation and related mathematical equations of those estimates. A selected list of estimates is described in Section 4, and their values in conducted experiments are displayed in Section 5.

## 4. Factors of Interest

As we focus on the linear system in scientific applications instead of random systems for benchmarking, we have less control over the specific values of matrix properties like condition numbers. Our interest is more on the number of clock qubits $n_c$ in the HHL circuit, which controls the accuracy of estimated eigenvalues. The error in eigenvalue estimation affects the solution of the linear system through Equation (2). From [35], to obtain an eigenvalue with $2^{-b}$ accuracy with at least $1 - p_{QPE,fail}$ success probability using QPE, we need

$$n_c = b + \left\lceil \log_2 \left( 2 + \frac{1}{2p_{QPE,fail}} \right) \right\rceil.$$

In the Qiskit-based HHL implementation that we used [55], it is suggested that

$$n_c = \max(n_d + 1, \lceil \log_2(\kappa + 1) \rceil) + \mathbb{I}_{nv} \tag{3}$$

where $\mathbb{I}_{nv} = 1$ if the coefficient matrix has a negative eigenvalue; otherwise, it is 0. In this paper, we will adjust $n_c$ to illustrate the influence of the QPE resources on the HHL circuit's total cost and the algorithm's accuracy in domain applications.

When discussing resource estimation under a fault-tolerant setting, our primary concerns are the estimated runtime, the number of physical qubits, and extra resources required from the QEC code. We adopt a distance-7 surface code that encodes 98 physical qubits into a single logical qubit. The theoretical logical qubit error rate is $3 \times 10^{-10}$, and the error correction threshold is 0.01. The Azure Quantum resource estimator provides several qubit parameter sets to simulate different qubit properties. The preset qubit settings we used in this paper are $(\text{ns}, 10^{-4})$ and $(\mu\text{s}, 10^{-4})$ from [42], where the former is close to the specifications of superconducting transmon qubits or spin qubits, and the latter is more relevant for trapped-ion qubits [42]. A list of detailed configurations of qubit parameter set $(\text{ns}, 10^{-4})$ and $(\mu\text{s}, 10^{-4})$ is in Table 3. We enforce 2-D nearest-neighbor connectivity of the qubits to simulate the connectivity constraint on real quantum computers. So we also demonstrate the changes in some factors before this constraint is enforced ("pre-layout") and after this constraint is enforced ("after layout").

**Table 3.** Qubit parameter configurations.

| | **Qubit Parameter Set** | **(ns, $10^{-4}$)** | **(µs, $10^{-4}$)** |
|---|---|---|---|
| Operation Time | Measurement | 100 ns | 100 µs |
| | Single-qubit gate | 50 ns | 100 µs |
| | Two-qubit gate | 50 ns | 100 µs |
| | $T$ gate | 50 ns | 100 µs |
| Error rate | Measurement | $10^{-4}$ | $10^{-4}$ |
| | Single-qubit gate | $10^{-4}$ | $10^{-4}$ |
| | Two-qubit gate | $10^{-4}$ | $10^{-4}$ |
| | $T$ gate | $10^{-4}$ | $10^{-6}$ |

Another important tunable parameter is the overall allowed errors for the algorithm, namely error budget. Its parameter value is equally divided into three parts:

- Logical error probability: the probability of at least one logical error;
- T-distillation error probability: the probability of at least one faulty T-distillation;
- Rotation synthesis error probability: the probability of at least one failed rotation synthesis.

There are also specific breakdowns in the resource required by QEC that are of interest [42,43]. We list them in Table 4.

**Table 4.** Factors of interest in QEC.

| Factors of Interest | Description |
|---|---|
| Number of logical qubits pre- and after layout | Under the nearest-neighbor constraint, extra logical qubits could be required to satisfy the connectivity needed in the algorithm (circuit); the relation is $n_{\text{after}} = 2n_{\text{alg}} + \left\lceil \sqrt{8n_{\text{alg}}} \right\rceil + 1$, where $n_{\text{alg}}$ is the number of logical qubits pre-layout and $n_{\text{after}}$ is the number of qubits after layout |
| Number of physical qubits for the algorithm | The product of the number of logical qubits after layout and the number of physical qubits needed to encode one logical qubit |
| Number of physical qubits for $T$ factories | $T$ factories produce $T$ states to implement non-Clifford operations in a circuit |
| Number of physical qubits | The sum of the number of physical qubits for the algorithm and the number of physical qubits for $T$ factories |
| Number of $T$ states | The estimator requires 1 $T$ state for each of the $T$ gates in a circuit, 4 $T$ states for each of the $CCZ$ and $CCiX$ gates, and 18 $T$ states for each of the arbitrary single-qubit rotation gates |
| Number of $T$ factories | Determined from algorithm runtime, $T$ state per $T$ factory, the number of $T$ states, and $T$ factor duration through the equation $\left\lceil \frac{T \text{ state} \cdot T \text{ factor duration}}{T \text{ state per } T \text{ factory} \cdot \text{algorithm runtime}} \right\rceil$ |
| Number of logical cycles for the algorithm | The logical depth of the algorithm |
| Min. logical qubit error rate required to run the algorithm within the error budget | $\frac{\text{logical error probability}}{\text{Number of logical qubits} \cdot \text{Number of logical cycles}}$ |
| Min. $T$ state error rate required for distilled $T$ states | $\frac{T \text{ distillation error probability}}{\text{Total number of } T \text{ states}}$ |

## 5. Scientific Applications and Evaluation

This section examines the utilization of the HHL algorithm in the fields of power grids and climate projection. We evaluate the performance of HHL in terms of solution accuracy, resource cost, and influence on convergence speed for applicable problems.

In addition to the hardware specifications in Section 3, all resource estimator jobs are run on the Azure Quantum cloud server. Due to the limitation on the cloud service usage, we cannot examine some of the deepest circuits in this section with the resource estimator, and all evaluated circuits are transpiled.

with respect to a given basis gate set from the estimator using the transpiler in Qiskit. The optimization level of the transpiler is set to level 2. The Qiskit version is 0.46. The Azure Quantum version is 0.30.0. The MATPOWER version is 7.1.

### *5.1. Power Flow Problem in Power Grid*

The use of quantum algorithms has drawn much attention in recent research on power system applications, especially the areas where quantum linear system solver can be deployed, including power flow, contingency analysis, state estimation, and transient simulation [60–65]. The specific problem type we illustrated in this section is an alternating current power flow problem.

The power flow equations are essential to analyzing the steady-state behavior of power systems by describing the relationship between bus voltages (magnitude and phase angles), currents, and power injections in a power system. The basic power flow equations are as follows:

$$P_k = \sum_{j=1}^{n} \Big( |V_k||V_j| \text{Re}(Y_{kj}^*) \cos(\theta_{kj}) + |V_k||V_j| \text{Im}(Y_{kj}^*) \sin(\theta_{kj}) \Big)$$

$$Q_k = \sum_{j=1}^{n} \Big( |V_k||V_j| \text{Re}(Y_{kj}^*) \sin(\theta_{kj}) - |V_k||V_j| \text{Im}(Y_{kj}^*) \cos(\theta_{kj}) \Big)$$

where

- $P_k$: real power injection at bus $k$.
- $Q_k$: reactive power injection at bus $k$.
- $|V_k|$: voltage magnitude at bus $k$.
- $\theta_{kj}$: phase angle difference between bus $k$ and bus $j$.
- $Y_{kj}$: admittance between bus $k$ and bus $j$.

For a power flow problem with $B$ buses and $G$ generators, there are $2(B-1)-(G-1)$ unknowns representing voltage magnitudes, $|V_k|$, and phase angles, $\theta_k$, for load buses and voltage phase angles for generator buses. With the knowledge of the admittance matrix of the system that represents the nodal admittance of the buses, we can use the Newton–Raphson (N-R) method to solve power flow equation iteratively: after an initial guess for the voltages at all buses, in each N-R iteration, we solve

$$\begin{bmatrix} \frac{\partial \Delta \vec{P}}{\partial \vec{\theta}} & \frac{\partial \Delta \vec{P}}{\partial |\vec{V}|} \\ \frac{\partial \Delta \vec{Q}}{\partial \vec{\theta}} & \frac{\partial \Delta \vec{Q}}{\partial |\vec{V}|} \end{bmatrix} \begin{bmatrix} \Delta \vec{\theta} \\ \Delta |\vec{V}| \end{bmatrix} = - \begin{bmatrix} \Delta \vec{P} \\ \Delta \vec{Q} \end{bmatrix} \tag{4}$$

where $\Delta P_k$ and $\Delta Q_k$ are computed using the admittance matrix, nodal power balance equation, and mismatch equations with the data from the last iteration or initial guess. Then, $\vec{\theta}$ and $|\vec{V}|$ are updated by $\Delta\vec{\theta}$ and $\Delta|\vec{V}|$, respectively. The algorithm is considered converged when $\|\Delta\vec{P}\|$ and $\|\Delta\vec{Q}\|$ are smaller than a convergence tolerance.

It is worth noting that while HHL can solve Equation (4) for the normalized solution state $[\Delta\vec{\theta}^T \ \Delta|\vec{V}|^T]^T$ with limited accuracy, the un-normalized vector could have a smaller norm than the accuracy of HHL. Thus, the final accuracy of voltage magnitude and phase angles is much higher than the accuracy used in HHL. This situation is similar to iterative refinement in semi-definite optimization in [13].

#### 5.1.1. Settings of the Numerical Experiments

The test case is the four buses and two generators problem in ([66], p. 377), coded in a MATLAB package called MATPOWER [67]. Based on the framework built in [68], we incorporate HHL circuits and quantum simulators into the solving process in MATPOWER. The linear systems of our interest are all $5 \times 5$ systems but not Hermitian. So, the actual input system is first expanded to $8 \times 8$ so the size of the RHS vector is the power of 2, and then it is enlarged to $16 \times 16$ following Equation (1). So, we eventually use 4 qubits to encode the vector $\vec{b}$. This process is illustrated in Figure 3a.

The default value of $n_c$ set by [35] using Equation (3) is 6. To demonstrate how the accuracy of eigenvalues affects an iterative algorithm, we select $n_c$ from 4 to 7. With 4 clock qubits in QPE and an ancillary qubit required by the HHL algorithm, the number of qubits in each HHL circuit ranges from 9 to 12. The N-R method converges when

$$\left\| \begin{bmatrix} \Delta \vec{P} \\ \Delta \vec{Q} \end{bmatrix} \right\|_\infty < 10^{-8}.$$

However, because the linear system formed in an N-R iteration depends on the solution from the previous N-R iteration, the linear systems at Iteration $j$ with different $n_c$ will differ. Our comparison focuses on the convergence speed and the final solution at the convergence instead of errors at each iteration across different $n_c$.

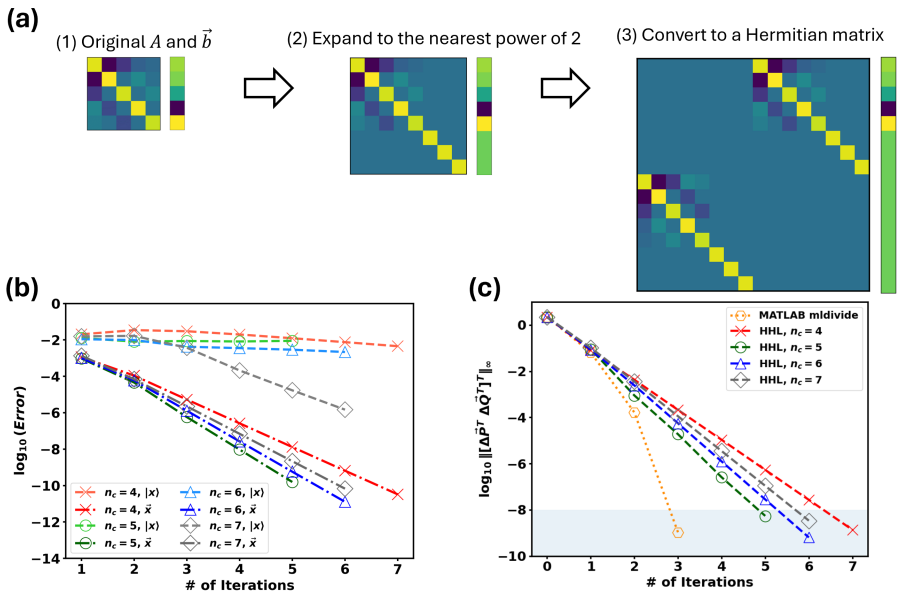


**Figure 3.** Matrix expansion and error plot for the experiments of the power flow problem. (**a**) Expanding the coefficient matrices and the RHS vector in Equation (4) to fulfill the requirements of the HHL algorithm. (**b**) Two types of solution errors in each N-R iteration with various values of $n_c$: the errors of the solution state (labeled "$|x\rangle$") in log base 10 , $\| \, |x\rangle - |x\rangle_{HHL} \, \|_2$, and the errors of the solutions (labeled "$\vec{x}$") in log base 10 , $\|\vec{x} - \vec{x}_{HHL}\|_2$. The symbols $A$, $\vec{b}$, $\vec{x}$, and $|x\rangle$ refer to the corresponding part in Equation (4). (**c**) The convergence performance under different values of $n_c$. The $y$-axis is the convergence criteria, the infinity norms of $[\Delta \vec{P}^T \ \Delta \vec{Q}^T]^T$, in log base 10. The value at iteration 0 represents the norm from the initial guess, and the gray-shaded area is where the convergence criteria are satisfied.

#### 5.1.2. Performance Evaluations

The sparsity of all tested coefficient matrices is 84.375% after the expansion, with condition numbers in the range of $[5.950, 5.970]$. The minimums of the magnitude of eigenvalues are in the range of $[12.263, 12.506]$, and the maximums are $[73.209, 74.659]$. Figure 3b,c provide illustrative evidence of the use of a less precise linear solver in the iterative method like the N-R method. Although the N-R method with an HHL subroutine converges slower than a classical linear solver in MATLAB, all methods converge under the same criteria and obtain a similar solution. A trade-off between convergence speed and complexity of linear system solving exists in our experiments.

On the other hand, if we compare the values of normalized error $\| \, |x\rangle - |x\rangle_{HHL} \, \|_2$, when $n_c = 4, 5, 6, 7$, using more clock qubits indeed leads to lower error from the HHL algorithm itself. However, increasing $n_c$ does not imply less error on the solution vectors, $\vec{x}_{HHL}$, nor faster convergence by looking at the values of $\|\vec{x} - \vec{x}_{HHL}\|_2$ and $[\Delta \vec{P}^T \ \Delta \vec{Q}^T]^T$ in Figure 3. The HHL algorithm with $n_c = 5$ gives the fastest convergence, which is smaller than the default value, 6, from Equation (3).

#### 5.1.3. Gate Counts and Depths of HHL Circuits.

Because the circuits from later iterations are in a similar resource demand, we only look at the circuits in the first iteration. The depths and gate counts of HHL circuits are the same across N-R iterations when $n_c$ is fixed. While HHL with $n_c = 5, 6, 7$ gives similar convergence speed and accuracy, the required resources to run the circuits exponentially increase as $n_c$ increases based on Table 5. On the other hand, although gate fusion employed in NWQSim does not mitigate these exponential trends, it maintains a constant proportional performance across various HHL circuits: a 79% reduction in gate counts on all tested circuits regardless of the value of $n_c$.

**Table 5.** Depths and gate counts of HHL circuits for power flow problems at Iteration 1.

| $n_d$ | $n_c$ | Depth | Gates | 2-Qubit Gates | Gates After Fusion | Reduction from Fusion |
|---|---|---|---|---|---|---|
| 4 | 4 | 65,824 | 86,262 | 30,651 | 18,060 | 79.06% |
| 4 | 5 | 135,986 | 178,180 | 63,315 | 37,283 | 79.08% |
| 4 | 6 | 276,308 | 361,980 | 128,631 | 75,717 | 79.08% |
| 4 | 7 | 556,950 | 729,534 | 259,247 | 152,570 | 79.09% |

#### 5.1.4. Resource Estimation in a Fault-Tolerant Scenario

Encoded by the surface code described in Section 4 along with a nearest-neighbor connectivity constraint, we estimate the runtime of HHL circuits by the Azure Quantum resource estimator and summarize the data in Figure 4. A strong and consistent linear correlation between the number of clock qubits in QPE, $n_c$, and the runtime in log base 10 is displayed across qubit parameter sets and error budgets. Every extra clock qubit brings $10^{0.322} \approx 2.099$ times longer runtime when the error budget is 0.1 and $10^{0.375} \approx 2.371$ times longer when the error budget is 0.01. This multiplier shows an increasing trend when the error budget decreases. Similar correlations are also demonstrated in Figure 5a,b when we further investigate how $n_c$ affects the number of logical cycles for the circuit and the number of $T$ states. Generally, the exponential dependencies of runtime, number of logical cycles, and number of $T$ states on $n_c$ match the relationship between the number of gates in HHL circuits and $n_c$. Note that the slopes of the fitted line in Figure 5a,b are not sensitive to error budgets, different from the behavior in Figure 4. Error budgets affect the constant multiplier of the growth of logical cycles and the number of $T$ states more.

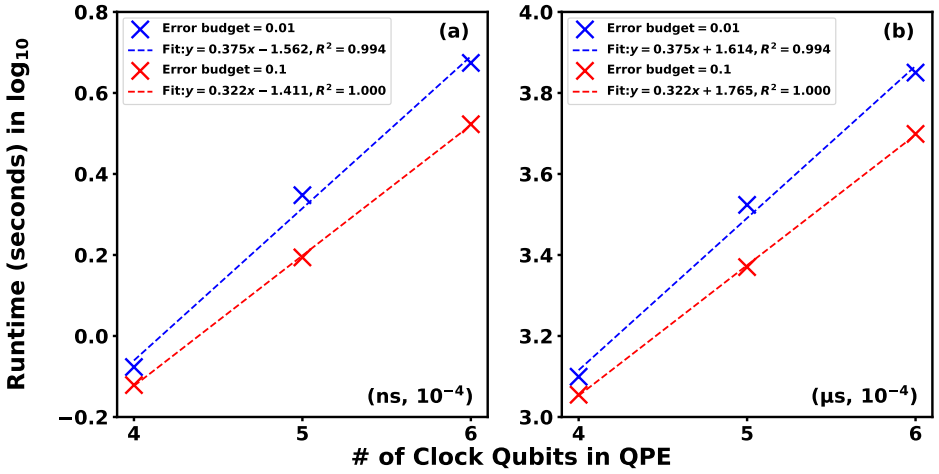


**Figure 4.** The runtime in seconds as a function of the number of clock qubits in QPE under the qubit parameter set (**a**) (ns, $10^{-4}$) and (**b**) (μs, $10^{-4}$). The estimated circuits are HHL circuits for power flow problems.

Table 6 summarizes the other factors of our interest. Those factors have the same values in (ns, $10^{-4}$) and (μs, $10^{-4}$) settings. Note that there is a dramatic fall in the number of physical qubits when the error budget is 0.01 and $n_c$ raises from 4 to 5. Combined with Figure 5c,d, this reduction comes from a large drop in the number of physical qubits spent on $T$ factories, a dominant demand on physical qubits instead of the quantum algorithm itself. The circuit requires 15 $T$ factories when the error budget is 0.01 and $n_c = 4$, but this number is reduced to 12 when $n_c = 5$. Recall the definition of the number of $T$ factories in Table 4, based on the fitted coefficients in Figures 4 and 5; we can see while the increase in $n_c$ from 4 to 5 leads to $10^{0.322}$ times more $T$ states, the runtime becomes $10^{0.375}$ times larger. Since $T$ factory duration and $T$ states per factory are kept constant, the faster-growing runtime reduces the number of $T$ factories required, thus decreasing the overall number of

physical qubits required. This phenomenon does not occur when the error budget is 0.1 because the growth of runtime and $T$-state count are at the same speed.

**Table 6.** Factors of interest for fault-tolerant HHL circuits in power flow problems.

| Error Budget | $n_c$ | Physical Qubits After Layout | Logical Qubits Pre- and After Layout | Min. Logical Qubit Error Rate | Min. $T$ State Error Rate |
|---|---|---|---|---|---|
| 0.01 | 4 | 32,144 | 9 to 28 | $3.977 \times 10^{-10}$ | $9.831 \times 10^{-9}$ |
| | 5 | 28,380 | 10 to 30 | $1.797 \times 10^{-10}$ | $4.762 \times 10^{-9}$ |
| | 6 | 28,866 | 11 to 33 | $7.700 \times 10^{-11}$ | $2.236 \times 10^{-9}$ |
| 0.1 | 4 | 32,144 | 9 to 28 | $4.406 \times 10^{-9}$ | $1.098 \times 10^{-7}$ |
| | 5 | 32,340 | 10 to 30 | $1.990 \times 10^{-9}$ | $5.319 \times 10^{-8}$ |
| | 6 | 32,634 | 11 to 33 | $8.487 \times 10^{-10}$ | $2.483 \times 10^{-8}$ |

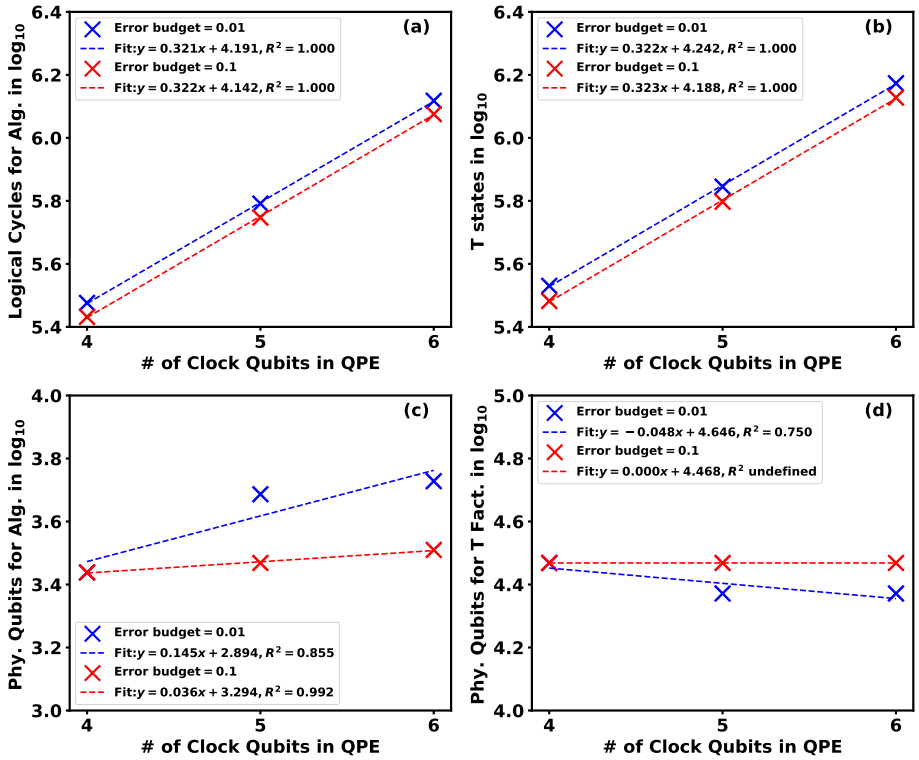


**Figure 5.** The number of (**a**) logical cycles for the algorithm, (**b**) $T$ states, (**c**) physical qubits for the algorithm after layout, and (**d**) physical qubits for the $T$ factories as functions of the number of clock qubits in QPE, respectively. The estimated circuits are HHL circuits for power flow problems. Both qubit parameter sets (ns, $10^{-4}$) and (µs, $10^{-4}$) have the same values under the same error budget for all four factors in the plots.

### *5.2. Heat Transfer Problem in Climate Projection*

Linear solvers are deeply embedded in linear or non-linear differential equation solving through numerical methods such as the Carleman linearization and the finite difference method [69]. Such methods discretize the domain of the problems into grids, and the dimension of the formed linear system scales as the size of discretization. The number of grid points scales polynomially with system size, while the demands for solving such differential equations (DEs) are ubiquitous in science and engineering. Due to the exponential speedup in problem dimension, the combination of quantum linear solvers and these numerical methods has become an attractive direction [69–72]. For example, accurate climate projection, one of the most scientifically challenging and socially urgent problems, is cursed by high dimension and could be revolutionized by quantum computing. In this

section, we explore the application of a quantum linear system solver to the heat transfer equation that is important for atmospheric processes related to climate projection.

#### 5.2.1. Settings of the Numerical Experiments

In this section, we examine the two-dimensional (2-D) heat diffusion equation in [73]

$$\frac{\partial T}{\partial t} = D\nabla^2 T + F \tag{5}$$

where $T$ represents the temperature at a given 2-D point and time, $D$ is the heat transfer coefficient, and $F$ is the forcing term consisting of arbitrary boundary and initial conditions. Equation (5) is a linear partial differential equation. We discretize Equation (5) in space and time into a system of ordinary differential equations using the finite difference method,

$$AT = F, \tag{6}$$

where $A$ is the resulting coefficient matrix. Take the square lattices with a lateral size of three grid points and five grid points, and the resulting dimension of $A$ is $9 \times 9$ and $25 \times 25$, respectively. Such configurations require 4 qubits and 5 qubits to represent the RHS vectors ($F$ term in Equation (6)) in both linear systems, respectively. Let $A^{(heat,l)}$ be the coefficient matrix generated from $l$ number of grid points; the entry values are

$$A_{pq}^{(heat,l)} = \begin{cases} 1+4r, & p = q \\ -r & p = q \pm 1 \text{ or } \text{ or } p = q \pm l \\ 0, & \text{otherwise} \end{cases}$$

where $p$ and $q$ denote the index of the entries of $A$, and $r$ is 0.00016 in the three-point case and 0.00064 in the five-point case.

#### 5.2.2. Performance and Resource Evaluations

The coefficient matrices are Hermitian by design, so we only need to expand the dimension to the nearest power of 2, i.e., 16 and 32. After dimension expansion, the coefficient matrices have sparsity 82.81% and 88.28%, respectively. Both matrices have condition number 1, and all of their eigenvalues are around 1.

When $n_d = 4$, gate counts in Tables 5 and 7 have almost the same numbers of circuit depths and gate counts. However, if we compare across different $n_d$ in Table 7, significant increases appear in depths and all gate counts. This situation reflects one of Aaronson's concerns in [74] about the efficiency and the cost of data reading in quantum linear solvers. Furthermore, similar to the scenario in Section 5.1, the incremental of $n_c$, despite being very costly, has a limited contribution towards reducing errors, as shown in Figure 6.

**Table 7.** Depths and gate counts of HHL circuits for heat transfer problems.

| Dim. | $n_d$ | $n_c$ | Depth | # of Gates | # of 2-Qubit Gates | # of Gates After Fusion | Reduction from Fusion |
|---|---|---|---|---|---|---|---|
| $9 \times 9$ | 4 | 3 | 30,742 | 40,290 | 14,315 | 8445 | 79.04% |
| $9 \times 9$ | 4 | 4 | 65,824 | 86,262 | 30,651 | 18,061 | 79.06% |
| $9 \times 9$ | 4 | 5 | 135,986 | 178,180 | 63,315 | 37,284 | 79.08% |
| $9 \times 9$ | 4 | 6 | 276,308 | 361,980 | 128,631 | 75,718 | 79.08% |
| $25 \times 25$ | 5 | 3 | 133,966 | 175,253 | 62,546 | 37,147 | 78.80% |
| $25 \times 25$ | 5 | 4 | 287,134 | 375,643 | 134,046 | 79,547 | 78.82% |
| $25 \times 25$ | 5 | 5 | 593,948 | 777,069 | 277,230 | 164,338 | 78.85% |

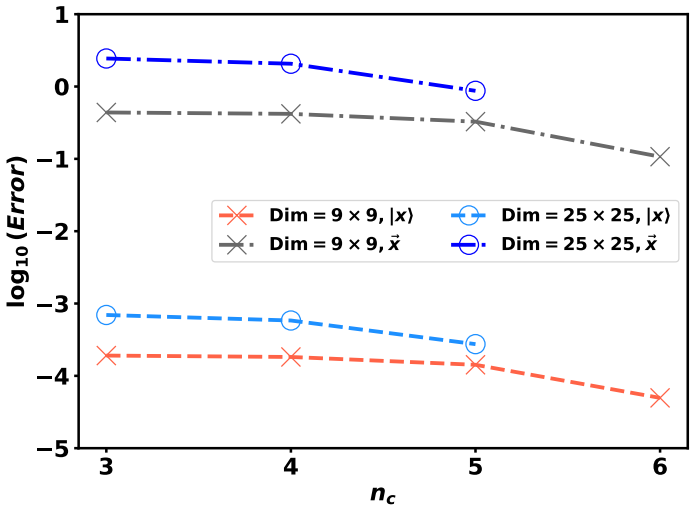


**Figure 6.** The errors of the solution states in log base 10 (labeled as "$|x\rangle$"), $\| \, |x\rangle - |x\rangle_{HHL} \, \|_2$, and the errors of the solution vectors in log base 10 (labeled as "$\vec{x}$"), $\|\vec{x} - \vec{x}_{HHL}\|_2$, are presented as functions of $n_c$ for two different numbers of grid points. The symbols $\vec{x}$ and $|x\rangle$, respectively, refer to the solution vector and the normalized solution vector.

#### 5.2.3. Resource Estimation in a Fault-Tolerant Scenario

Most of the observations from Figures 7 and 8 and Table 8 for both problem sizes are analogous to the findings in Section 5.1.4, including the numerical values of the fitted-line coefficients related to runtime, logical cycles, and the number of $T$ states. The significant influence brought by deeper data-loading modules for the five-point problem is parallel shifts on longer runtime, more logical cycles, more $T$ states, and more strict requirements on the logical qubit error rate and $T$ state error rate. More data-loading qubits do not affect the growth speed of the logical cycle and the number of $T$ states. Due to the limitation of computational time in Azure Quantum cloud service, we cannot collect more data points to understand this correlation better. However, from a theoretical perspective, this is expected because the QPE costs of HHL circuits are the same with the same $n_c$ in the power flow and heat transfer problems.

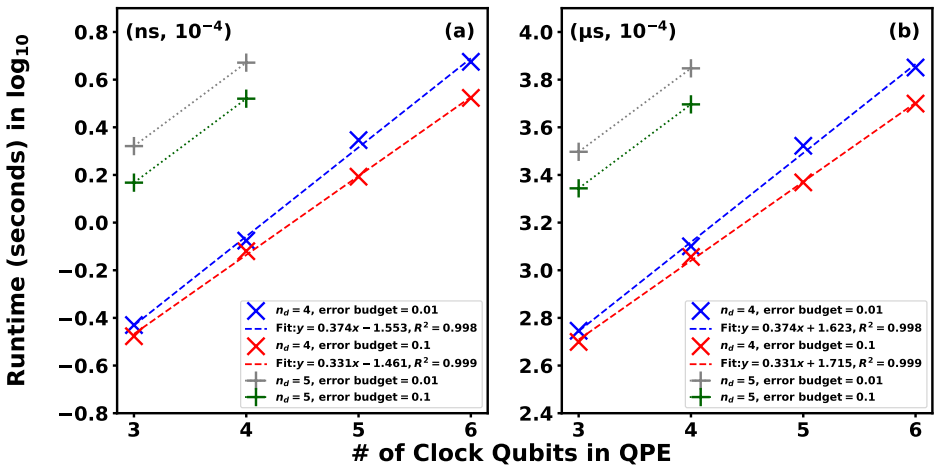


**Figure 7.** The runtime in seconds as a function of the number of clock qubits in QPE under the qubit parameter set (**a**) (ns, $10^{-4}$) and (**b**) (μs, $10^{-4}$). The estimated circuits are HHL circuits for the heat transfer problem.

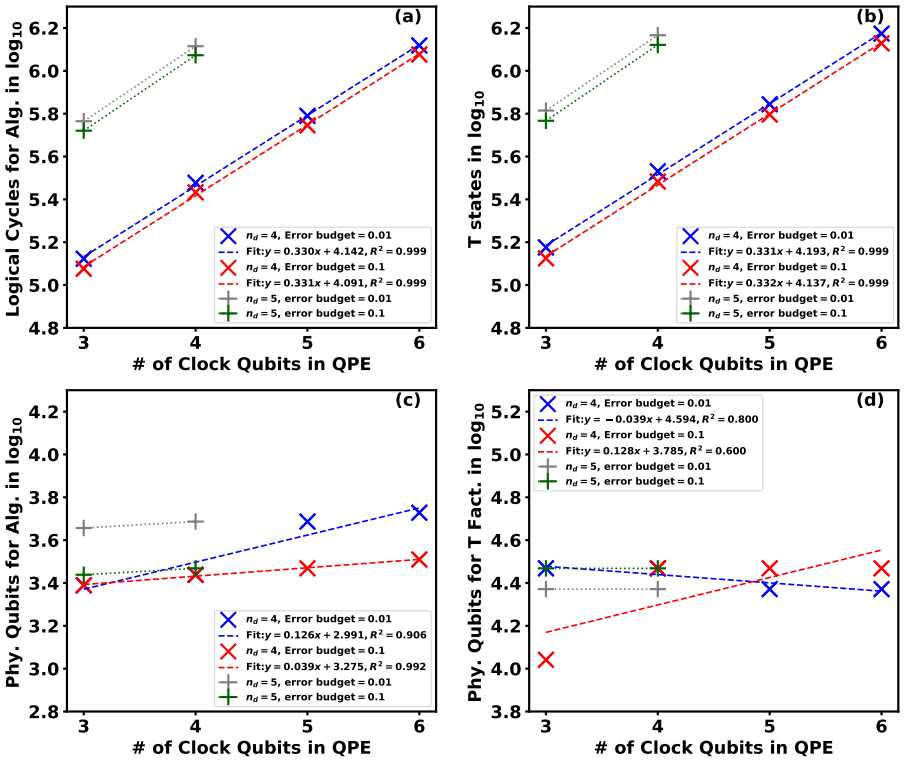


**Figure 8.** The number of (**a**) logical cycles for the algorithm, (**b**) $T$ states, (**c**) physical qubits for the algorithm after layout, and (**d**) physical qubits for the $T$ factories as functions of the number of clock qubits in QPE, respectively. The estimated circuits are HHL circuits for the heat transfer problem. Both qubit parameter sets (ns, $10^{-4}$) and (µs, $10^{-4}$) have the same values under the same error budget for all four factors in the plots.

**Table 8.** Factors of interest for fault-tolerant HHL circuits in heat transfer problems.

| Error Budget | $(n_d, n_c)$ | Physical Qubits After Layout | Logical Qubits Pre- and After Layout | Min. Logical Qubit Error Rate | Min. $T$ State Error Rate |
|---|---|---|---|---|---|
| 0.01 | (4, 3) | 31,850 | 8 to 25 | $1.01 \times 10^{-9}$ | $2.22 \times 10^{-8}$ |
| | (4, 4) | 32,144 | 9 to 28 | $3.97 \times 10^{-10}$ | $9.81 \times 10^{-9}$ |
| | (4, 5) | 28,380 | 10 to 30 | $1.80 \times 10^{-10}$ | $4.77 \times 10^{-9}$ |
| | (4, 6) | 28,866 | 11 to 33 | $7.69 \times 10^{-11}$ | $2.23 \times 10^{-9}$ |
| | (5, 3) | 28,056 | 9 to 28 | $2.05 \times 10^{-10}$ | $5.11 \times 10^{-9}$ |
| | (5, 4) | 28,380 | 10 to 30 | $8.53 \times 10^{-11}$ | $2.27 \times 10^{-9}$ |
| 0.1 | (4, 3) | 13450 | 8 to 25 | $1.12 \times 10^{-8}$ | $2.50 \times 10^{-7}$ |
| | (4, 4) | 32,144 | 9 to 28 | $4.40 \times 10^{-9}$ | $1.10 \times 10^{-7}$ |
| | (4, 5) | 32,340 | 10 to 30 | $2.00 \times 10^{-9}$ | $5.33 \times 10^{-8}$ |
| | (4, 6) | 32,634 | 11 to 33 | $8.47 \times 10^{-10}$ | $2.48 \times 10^{-8}$ |
| | (5, 3) | 32,144 | 9 to 28 | $2.27 \times 10^{-9}$ | $5.70 \times 10^{-8}$ |
| | (5, 4) | 32,340 | 10 to 30 | $9.39 \times 10^{-10}$ | $2.52 \times 10^{-8}$ |

## 6. Discussion

This paper evaluates and analyzes the performance and resources required for the HHL algorithm in various scientific and engineering problems. There are still multiple points we need to address in future work. The foremost limitation in this work is the data-loading module in the HHL circuit generation. While the data-loading algorithm in [55] can encode an arbitrary vector into a quantum circuit, the circuit depth of this module is exponential in the number of qubits. Thus, this first part of the circuit severely damages the potential quantum speedup from HHL. We mitigate this drawback by comparing

the outcomes from problems of different sizes to isolate the influence of the data-loading module. An important future direction is incorporating an efficient data-loading scheme into our analysis framework, like block-encoding in [8]. A different data-loading method could have a different accuracy, so it is necessary to investigate how data-loading accuracy and condition number of coefficient matrices collectively affect the solution accuracy. This future direction illustrates the second drawback of this study. That is, our tested coefficient matrices are all well-conditioned. Because our experiments do not utilize randomly generated test cases, we have less control over the matrix properties, including condition number and sparsity. A potential source of ill-conditioned test cases is the methods that naturally have ill-conditioned matrices, such as the Newton systems produced by the interior-point method in optimization problems [13]. Thus, to solve those systems, iterative refinement with the HHL algorithm [9] and a variant of the HHL algorithm in [15], accompanied by the sparse approximate inverse preconditioner, is in our outlook. Limited by the single-job running time in the Azure Quantum cloud server, we cannot process large HHL circuits, mainly limited by the number of gates. This restricts the number of clock qubits in the QPE and the number of data points in each plot in Section 5. This is why we only discuss the correlations whose coefficients of determination are almost 1. In future studies, we will dismantle the whole HHL circuit into different modules and evaluate the resource cost separately.

Some additional research can be conducted to further enhance our understanding of the application of quantum algorithms in scientific problems. An important direction is understanding the implication of various noise models on the HHL algorithm. We plan to conduct those experiments with the high-precision noise simulator in [75]. We can also include the quantum algorithms that address similar scientific applications into our resource analysis framework, such as quantum differential equation solvers in [76,77] and quantum optimizers in [78,79].

## 7. Conclusions

In this paper, we investigate the practical applications and scalability of the HHL algorithm in solving quantum linear systems associated with scientific problems like power grids and heat transfer problems. Through the NWQSim package on high-performance computing platforms, we highlight the benefits of the utilization of low-accuracy QPE in HHL for both iterative and non-iterative methods in practice: low-accuracy QPE can exponentially reduce the gate counts and circuit depth in an HHL circuit, while keeping the same solution accuracy in iterative methods like the Newton–Raphson method and maintaining a similar level of accuracy in a non-iterative method like the finite difference method.

Furthermore, with the Azure Quantum resource estimator, we evaluate the resource requirements of HHL circuits in our experiments under two settings that simulate superconducting and trapped-ion qubits. The correlations between QEC-related criteria and the input HHL circuits have been thoroughly studied. The runtime, number of logical cycles, and number of $T$ states have exponential dependencies on the number of clock qubits in QPE. However, this relation is not necessarily inherited by the number of physical qubits demanded. In our experiments, we find that even as $n_c$ increases and the error budget reduces, it is possible that $T$ factory demand also decreases. More specifically, if the runtime growth is faster than the required amount of $T$ states, the circuit needs fewer $T$ factories and thus fewer physical qubits to prepare $T$ factories. Since the growth of runtime is sensitive to the error budget, it is possible to reduce the physical qubit requirement if a low error budget is achievable on early fault-tolerant quantum devices.

Our study provides pivotal insights into the operational requirements of quantum linear system algorithms, paving the way for further empirical studies. We propose future

research on the applications of quantum linear system solvers and iterative refinement on high-fidelity quantum computers for small-scale experiments. For large-scale experiments, we suggest using noise-modeled simulators on high-performance platforms. In the context of QEC and early fault-tolerant quantum computing, we believe it is crucial to focus on controlling the resource cost of $T$ factories by considering the runtime and error budget. These research directions hold promise for bridging the gap between theoretical potential and practical usability in quantum computing. All the code in this research will be hosted in a public repository (https://github.com/pnnl/nwqlib, accessed on 4 August 2025).

**Author Contributions:** Conceptualization, M.Z., C.L., S.S., X.L., J.M., Y.C. and A.L.; methodology, M.Z., C.L., S.S. and A.L.; software, M.Z. and A.L.; validation, S.S., X.L., J.M., Y.C. and A.L.; formal analysis, M.Z. and C.L.; investigation, M.Z.; resources, X.L., J.M., Y.C. and A.L.; data curation, M.Z., X.L., J.M. and Y.C.; writing—original draft preparation, M.Z., C.L., S.S., X.L., J.M., Y.C. and A.L.; writing—review and editing, M.Z. and A.L.; visualization, M.Z.; supervision, S.S., J.M., Y.C. and A.L.; project administration, A.L.; funding acquisition, A.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by the Quantum Algorithms and Architecture for Domain Science Initiative (QuAADS), under the Laboratory Directed Research and Development (LDRD) Program at Pacific Northwest National Laboratory (PNNL). This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231. PNNL is a multi-program national laboratory operated for the U.S. Department of Energy (DOE) by Battelle Memorial Institute under Contract No. DE-AC05-76RL01830.

**Data Availability Statement:** The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author(s).

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Deutsch, D.; Jozsa, R. Rapid solution of problems by quantum computation. *Proc. R. Soc. London. Ser. A Math. Phys. Sci.* **1992**, *439*, 553–558. [CrossRef]
2. Shor, P. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, USA, 20–22 November 1994; pp. 124–134. [CrossRef]
3. Harrow, A.W.; Hassidim, A.; Lloyd, S. Quantum Algorithm for Linear Systems of Equations. *Phys. Rev. Lett.* **2009**, *103*, 150502. [CrossRef]
4. Rebentrost, P.; Lloyd, S. Quantum computational finance: Quantum algorithm for portfolio optimization. *arXiv* **2018**, arXiv:1811.03975.
5. Duan, B.; Yuan, J.; Yu, C.H.; Huang, J.; Hsieh, C.Y. A survey on HHL algorithm: From theory to application in quantum machine learning. *Phys. Lett. A* **2020**, *384*, 126595. [CrossRef]
6. Liu, J.; Liu, M.; Liu, J.P.; Ye, Z.; Wang, Y.; Alexeev, Y.; Eisert, J.; Jiang, L. Towards provably efficient quantum algorithms for large-scale machine-learning models. *Nat. Commun.* **2024**, *15*, 434. [CrossRef]
7. Liu, J.P.; øie Kolden, H.; Krovi, H.K.; Loureiro, N.F.; Trivisa, K.; Childs, A.M. Efficient quantum algorithm for dissipative nonlinear differential equations. *Proc. Natl. Acad. Sci. USA* **2021**, *118*, e2026805118. [CrossRef]
8. Casares, P.A.; Martin-Delgado, M.A. A quantum interior-point predictor–corrector algorithm for linear programming. *J. Phys. A Math. Theor.* **2020**, *53*, 445305. [CrossRef]
9. Mohammadisiahroudi, M.; Fakhimi, R.; Terlaky, T. Efficient use of quantum linear system algorithms in interior point methods for linear optimization. *arXiv* **2022**, arXiv:2205.01220.
10. Mohammadisiahroudi, M.; Wu, Z.; Augustino, B.; Carr, A.; Terlaky, T. Improvements to quantum interior point method for linear optimization. *arXiv* **2023**, arXiv:2310.07574.

11. Wu, Z.; Mohammadisiahroudi, M.; Augustino, B.; Yang, X.; Terlaky, T. An inexact feasible quantum interior point method for linearly constrained quadratic optimization. *Entropy* **2023**, *25*, 330.
12. Augustino, B.; Nannicini, G.; Terlaky, T.; Zuluaga, L.F. Quantum interior point methods for semidefinite optimization. *Quantum* **2023**, *7*, 1110.
13. Mohammadisiahroudi, M.; Augustino, B.; Sampourmahani, P.; Terlaky, T. Quantum Computing Inspired Iterative Refinement for Semidefinite Optimization. *arXiv* **2023**, arXiv:2312.11253.
14. Ambainis, A. Variable time amplitude amplification and quantum algorithms for linear algebra problems. In *29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012), Proceedings of the STACS 2012, Paris, France, 29 February–3 March 2012*; Christoph Dürr, T.W., Ed.; Schloss Dagstuhl–Leibniz-Zentrum für Informatik: Wadern, Germany, 2012; Volume 14, pp. 636–647.
15. Clader, B.D.; Jacobs, B.C.; Sprouse, C.R. Preconditioned quantum linear system algorithm. *Phys. Rev. Lett.* **2013**, *110*, 250504.
16. Childs, A.M.; Kothari, R.; Somma, R.D. Quantum Algorithm for Systems of Linear Equations with Exponentially Improved Dependence on Precision. *SIAM J. Comput.* **2017**, *46*, 1920–1950. [CrossRef]
17. Chakraborty, S.; Gilyén, A.; Jeffery, S. The power of block-encoded matrix powers: Improved regression techniques via faster Hamiltonian simulation. In Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019), Patras, Greece, 9–12 July 2019. [CrossRef]
18. Subaşı, Y.b.; Somma, R.D.; Orsucci, D. Quantum Algorithms for Systems of Linear Equations Inspired by Adiabatic Quantum Computing. *Phys. Rev. Lett.* **2019**, *122*, 060504. [CrossRef]
19. An, D.; Lin, L. Quantum Linear System Solver Based on Time-optimal Adiabatic Quantum Computing and Quantum Approximate Optimization Algorithm. *ACM Trans. Quantum Comput.* **2022**, *3*, 5. [CrossRef]
20. Costa, P.C.; An, D.; Sanders, Y.R.; Su, Y.; Babbush, R.; Berry, D.W. Optimal scaling quantum linear-systems solver via discrete adiabatic theorem. *PRX Quantum* **2022**, *3*, 040303. [CrossRef]
21. Vazquez, A.C.; Hiptmair, R.; Woerner, S. Enhancing the quantum linear systems algorithm using Richardson extrapolation. *ACM Trans. Quantum Comput.* **2022**, *3*, 1–37.
22. Jennings, D.; Lostaglio, M.; Pallister, S.; Sornborger, A.T.; Subaşı, Y. Efficient quantum linear solver algorithm with detailed running costs. *arXiv* **2023**, arXiv:2305.11352.
23. Yalovetzky, R.; Minssen, P.; Herman, D.; Pistoia, M. Hybrid HHL with Dynamic Quantum Circuits on Real Hardware. *arXiv* **2021**, arXiv:2110.15958.
24. Sævarsson, B.; Chatzivasileiadis, S.; Jóhannsson, H.; Østergaard, J. Quantum Computing for Power Flow Algorithms: Testing on real Quantum Computers. In Proceedings of the 11th Bulk Power Systems Dynamics and Control Symposium (IREP 2022), Banff, AB, Canada, 25–30 July 2022.
25. Morgan, J.; Ghysels, E.; Mohammadbagherpoor, H. An Enhanced Hybrid HHL Algorithm. *arXiv* **2024**, arXiv:2404.10103.
26. Wen, J.; Kong, X.; Wei, S.; Wang, B.; Xin, T.; Long, G. Experimental realization of quantum algorithms for a linear system inspired by adiabatic quantum computing. *Phys. Rev. A* **2019**, *99*, 012320.
27. Preskill, J. Quantum Computing in the NISQ era and beyond. *Quantum* **2018**, *2*, 79. [CrossRef]
28. Krinner, S.; Lacroix, N.; Remm, A.; Di Paolo, A.; Genois, E.; Leroux, C.; Hellings, C.; Lazar, S.; Swiadek, F.; Herrmann, J.; et al. Realizing repeated quantum error correction in a distance-three surface code. *Nature* **2022**, *605*, 669–674.
29. Google Quantum AI. Suppressing quantum errors by scaling a surface code logical qubit. *Nature* **2023**, *614*, 676–681. [CrossRef]
30. Wang, Y.; Simsek, S.; Gatterman, T.M.; Gerber, J.A.; Gilmore, K.; Gresh, D.; Hewitt, N.; Horst, C.V.; Matheny, M.; Mengle, T.; et al. Fault-tolerant one-bit addition with the smallest interesting colour code. *arXiv* **2023**, arXiv:2309.09893.
31. Mayer, K.; Ryan-Anderson, C.; Brown, N.; Durso-Sabina, E.; Baldwin, C.H.; Hayes, D.; Dreiling, J.M.; Foltz, C.; Gaebler, J.P.; Gatterman, T.M.; et al. Benchmarking logical three-qubit quantum Fourier transform encoded in the Steane code on a trapped-ion quantum computer. *arXiv* **2024**, arXiv:2404.08616.
32. Bluvstein, D.; Evered, S.J.; Geim, A.A.; Li, S.H.; Zhou, H.; Manovitz, T.; Ebadi, S.; Cain, M.; Kalinowski, M.; Hangleiter, D.; et al. Logical quantum processor based on reconfigurable atom arrays. *Nature* **2024**, *626*, 58–65.
33. da Silva, M.; Ryan-Anderson, C.; Bello-Rivas, J.; Chernoguzov, A.; Dreiling, J.; Foltz, C.; Gaebler, J.; Gatterman, T.; Hayes, D.; Hewitt, N.; et al. Demonstration of logical qubits and repeated error correction with better-than-physical error rates. *arXiv* **2024**, arXiv:2404.02280.
34. Gupta, R.S.; Sundaresan, N.; Alexander, T.; Wood, C.J.; Merkel, S.T.; Healy, M.B.; Hillenbrand, M.; Jochym-O'Connor, T.; Wootton, J.R.; Yoder, T.J.; et al. Encoding a magic state with beyond break-even fidelity. *Nature* **2024**, *625*, 259–263.
35. Nielsen, M.A.; Chuang, I.L. *Quantum Computation and Quantum Information*; Cambridge University Press: Cambridge, UK, 2001; Volume 2.
36. Pascuzzi, V.R.; Bao, N.; Li, A. On the importance of scalability and resource estimation of quantum algorithms for domain sciences. *arXiv* **2022**, arXiv:2205.00585.

37. Fedorov, D.; Otten, M.; Kang, B.; Benali, A.; Habib, S.; Gray, S.; Alexeev, Y. Quantum Resource Estimation for Quantum Chemistry Algorithms. In Proceedings of the 2022 IEEE International Conference on Quantum Computing and Engineering (QCE), Broomfield, CO, USA, 18–23 September 2022; IEEE: New York, NY, USA, 2022; pp. 859–861.
38. Grassl, M.; Langenberg, B.; Roetteler, M.; Steinwandt, R. Applying Grover's algorithm to AES: Quantum resource estimates. In Proceedings of the 7th International Workshop on Post-Quantum Cryptography, Fukuoka, Japan, 24–26 February 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 29–43.
39. Gidney, C.; Ekerå, M. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum* **2021**, *5*, 433.
40. Roetteler, M.; Naehrig, M.; Svore, K.M.; Lauter, K. Quantum resource estimates for computing elliptic curve discrete logarithms. In Proceedings of the Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, 3–7 December 2017; Proceedings, Part II 23; Springer: Berlin/Heidelberg, Germany, 2017; pp. 241–270.
41. Scherer, A.; Valiron, B.; Mau, S.C.; Alexander, S.; Van den Berg, E.; Chapuran, T.E. Concrete resource analysis of the quantum linear-system algorithm used to compute the electromagnetic scattering cross section of a 2D target. *Quantum Inf. Process.* **2017**, *16*, 1–65.
42. Beverland, M.E.; Murali, P.; Troyer, M.; Svore, K.M.; Hoeffler, T.; Kliuchnikov, V.; Low, G.H.; Soeken, M.; Sundaram, A.; Vaschillo, A. Assessing requirements to scale to practical quantum advantage. *arXiv* **2022**, arXiv:2211.07629.
43. van Dam, W.; Mykhailova, M.; Soeken, M. Using Azure Quantum Resource Estimator for Assessing Performance of Fault Tolerant Quantum Computation. In Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis, Denver, CO, USA, 12–17 November 2023; pp. 1414–1419.
44. Li, A. NWQSim: Northwest Quantum Circuit Simulation Environment. 2024. Available online: https://github.com/pnnl/NWQ-Sim (accessed on 26 March 2024).
45. Zaman, A.; Morrell, H.J.; Wong, H.Y. A Step-by-Step HHL Algorithm Walkthrough to Enhance Understanding of Critical Quantum Computing Concepts. *IEEE Access* **2023**, *11*, 77117–77131.
46. Low, G.H.; Chuang, I.L. Hamiltonian simulation by qubitization. *Quantum* **2019**, *3*, 163.
47. Camps, D.; Van Beeumen, R. Fable: Fast approximate quantum circuits for block-encodings. In Proceedings of the 2022 IEEE International Conference on Quantum Computing and Engineering (QCE), Broomfield, CO, USA, 18–23 September 2022; IEEE: New York, NY, USA, 2022; pp. 104–113.
48. Camps, D.; Lin, L.; Van Beeumen, R.; Yang, C. Explicit quantum circuits for block encodings of certain sparse matrices. *SIAM J. Matrix Anal. Appl.* **2024**, *45*, 801–827.
49. Giovannetti, V.; Lloyd, S.; Maccone, L. Architectures for a quantum random access memory. *Phys. Rev. A* **2008**, *78*, 052310.
50. Kerenidis, I.; Prakash, A. Quantum recommendation systems. *arXiv* **2016**, arXiv:1603.08675.
51. Kerenidis, I.; Prakash, A. Quantum gradient descent for linear systems and least squares. *Phys. Rev. A* **2020**, *101*, 022316.
52. Liu, C.; Wang, M.; Stein, S.A.; Ding, Y.; Li, A. Quantum Memory: A Missing Piece in Quantum Computing Units. *arXiv* **2023**, arXiv:2309.14432.
53. Low, G.H.; Chuang, I.L. Optimal Hamiltonian Simulation by Quantum Signal Processing. *Phys. Rev. Lett.* **2017**, *118*, 010501. [CrossRef]
54. van Apeldoorn, J.; Cornelissen, A.; Gilyén, A.; Nannicini, G. Quantum tomography using state-preparation unitaries. In Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA),Florence, Italy, 22–25 January 2023; SIAM: Bangkok, Thailand, 2023; pp. 1265–1318.
55. Vazquez, A.C. Quantum Linear Solvers. 2022. Available online: https://github.com/anedumla/quantum_linear_solvers (accessed on 26 March 2024).
56. Iten, R.; Colbeck, R.; Kukuljan, I.; Home, J.; Christandl, M. Quantum circuits for isometries. *Phys. Rev. A* **2016**, *93*, 032318.
57. Li, A.; Fang, B.; Granade, C.; Prawiroatmodjo, G.; Heim, B.; Roetteler, M.; Krishnamoorthy, S. SV-sim: Scalable PGAS-based state vector simulation of quantum circuits. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, St. Louis, CA, USA, 14–19 November, 2021; pp. 1–14.
58. Qiskit Contributors. *Qiskit/qiskit-metapackage*, version 0.46.0. Qiskit: An Open-Source Framework for Quantum Computing; Zenodo: Geneva, Switzerland, 2024. [CrossRef]
59. Cirq Developers. *Cirq*, version v1.6.0. Zenodo: Geneva, Switzerland, 2023. [CrossRef]
60. Feng, F.; Zhou, Y.; Zhang, P. Quantum power flow. *IEEE Trans. Power Syst.* **2021**, *36*, 3810–3812.
61. Chen, Y.; Huang, Z.; Jin, S.; Li, A. Computing for power system operation and planning: Then, now, and the future. *iEnergy* **2022**, *1*, 315–324.
62. Zhou, Y.; Tang, Z.; Nikmehr, N.; Babahajiani, P.; Feng, F.; Wei, T.C.; Zheng, H.; Zhang, P. Quantum computing in power systems. *iEnergy* **2022**, *1*, 170–187.
63. Golestan, S.; Habibi, M.; Mousavi, S.M.; Guerrero, J.M.; Vasquez, J.C. Quantum computation in power systems: An overview of recent advances. *Energy Rep.* **2023**, *9*, 584–596.

64. Chen, Y.; Stein, S.; Li, A.; Huang, Z.H. Is It Coming Soon to Power Systems: Quantum Computing and Its Early Exploration. In Proceedings of the 2023 IEEE Power & Energy Society General Meeting (PESGM), Orlando, FL, USA, 16–20 July 2023; IEEE: New York, NY, USA, 2023; pp. 1–5.
65. Jing, H.; Li, Y.; Brandsema, M.J.; Chen, Y.; Yue, M. HHL algorithm with mapping function and enhanced sampling for model predictive control in microgrids. *Appl. Energy* **2024**, *361*, 122878.
66. Grainger, J.; Stevenson, W. *Power System Analysis*; McGraw-Hill Professional: New York, NY, USA, 1994.
67. Zimmerman, R.D.; Murillo-Sánchez, C.E. *MATPOWER*, version 7.1. Zenodo: Geneva, Switzerland, 2020. [CrossRef]
68. Zheng, M.; Chen, Y.; Yang, X.; Li, A. Early Exploration of a Flexible Framework for Efficient Quantum Linear Solvers in Power Systems. *arXiv* **2024**, arXiv:2402.08136.
69. Li, X.; Yin, X.; Wiebe, N.; Chun, J.; Schenter, G.K.; Cheung, M.S.; Mülmenstädt, J. Potential quantum advantage for simulation of fluid dynamics. *Phys. Rev. Res.* **2025**, *7*, 013036. [CrossRef]
70. Montanaro, A.; Pallister, S. Quantum algorithms and the finite element method. *Phys. Rev. A* **2016**, *93*, 032324.
71. Childs, A.M.; Liu, J.P.; Ostrander, A. High-precision quantum algorithms for partial differential equations. *Quantum* **2021**, *5*, 574.
72. Saha, K.K.; Robson, W.; Howington, C.; Suh, I.S.; Wang, Z.; Nabrzyski, J. Advancing Algorithm to Scale and Accurately Solve Quantum Poisson Equation on Near-term Quantum Hardware. *arXiv* **2022**, arXiv:2210.16668.
73. Li, X.Y.; Svensson, G.; Brandenburg, A.; Haugen, N.E. Cloud-droplet growth due to supersaturation fluctuations in stratiform clouds. *Atmos. Chem. Phys.* **2019**, *19*, 639–648.
74. Aaronson, S. Read the fine print. *Nat. Phys.* **2015**, *11*, 291–293. [CrossRef]
75. Li, A.; Liu, C.; Stein, S.; Suh, I.S.; Zheng, M.; Wang, M.; Shi, Y.; Fang, B.; Roetteler, M.; Humble, T. TANQ-Sim: Tensorcore Accelerated Noisy Quantum System Simulation via QIR on Perlmutter HPC. *arXiv* **2024**, arXiv:2404.13184.
76. An, D.; Liu, J.P.; Lin, L. Linear Combination of Hamiltonian Simulation for Nonunitary Dynamics with Optimal State Preparation Cost. *Phys. Rev. Lett.* **2023**, *131*, 150603. [CrossRef]
77. An, D.; Childs, A.M.; Lin, L. Quantum algorithm for linear non-unitary dynamics with near-optimal dependence on all parameters. *arXiv* **2023**, arXiv:2312.03916.
78. Leng, J.; Hickman, E.; Li, J.; Wu, X. Quantum hamiltonian descent. *arXiv* **2023**, arXiv:2303.01471.
79. Catli, A.B.; Simon, S.; Wiebe, N. Exponentially better bounds for quantum optimization via dynamical simulation. *arXiv* **2025**, arXiv:2502.04285.